

FIXED CLIENT IDENTIFICATION SYSTEM FOR  
POSITIVE IDENTIFICATION OF CLIENT TO SERVER

**CLAIM OF PRIORITY**

This application claims priority under 35 USC 119(e) to U.S. Patent  
5 Application Serial No. 60/421,285 filed on October 25, 2002, the entire contents of  
which are incorporated by reference.

**TECHNICAL FIELD**

This invention relates to security applications, and more particularly to  
identification of a user's computer.

10 **BACKGROUND**

Identification of a particular client computer system used for accessing a  
server is useful in secure applications where positive identification is desirable. In the  
past, systems for identifying client computers, browser cookies, for example, have  
had less than satisfactory capability of resisting tampering.

15 **SUMMARY**

A ClientID uniquely identifying a client machine is issued by the backend and  
stored on the client's machine upon first client application connection to the backend.  
On all subsequent connections, the client application retrieves the ClientID and sends  
it back to the backend. The ClientID mechanism includes features that make it very  
20 difficult for the user to remove or change the ClientID once it is set. In particular,  
according to the invention, this is accomplished by having the client application store  
at least two different scrambled versions of the ClientID in two separate locations in  
the client machine. Upon subsequent connection to the backend, the client application  
attempts to retrieve and unscramble the values at the two locations.

25 In the preferred embodiment, during the ClientID storage process, the backend  
generates a ClientID initially that contains a checksum and transmits it to the client  
application upon initial connection to the backend. The client application uses a first  
key to scramble the ClientID generating a first scrambled ClientID that is stored in the  
first predetermined location, for example the registry. A second key is used by the

client application to produce a second scrambled version of the ClientID that is stored in the second predetermined location, for example the system configuration file.

Upon subsequent connection of the backend, a retrieval process is invoked in which the client application retrieves the values at each location, unscrambles them  
5 using the respective keys, tests their checksums for verification, and compares the unscrambled values. If the checksums are both correct and the unscrambled values match, the retrieved ClientID is transmitted to the backend. Otherwise, the client application sends an appropriate error code to the backend.

The details of one or more embodiments of the invention are set forth in the  
10 accompanying drawings and the description below. Other features, objects, and advantages of the invention will be apparent from the description and drawings, and from the claims.

### DESCRIPTION OF DRAWINGS

FIG. 1 is a flow and block diagram of the ClientID storage process.

15 FIG. 2 is a flow diagram of the ClientID retrieval process.

Like reference symbols in the various drawings indicate like elements.

### DETAILED DESCRIPTION

ClientID is a special tag that uniquely identifies the client machine. Initially, the ClientID is generated by the backend and stored on client's machine upon first  
20 client application connection to the backend. On all subsequent connection, client application retrieves the ClientID and sends it back to the backend. Unlike browser cookies, the ClientID mechanism includes some special tamper-proof features that make it very difficult for the user to remove or change the ClientID once it is set.

Note: ClientID remains on the client's machine even after the client  
25 application is uninstalled. ClientID installation/retrieval occurs as a part of the client application startup process, as shown in FIG. 1, described in more detail below.

#### ***ClientID Storage Process***

ClientID is stored in at least two undisclosed locations on the client machine (for example, in the registry and system configuration file). As shown in FIG. 1, the

ClientID value is encrypted on the backend and contains a checksum. The client application has an ability to verify whether the checksum is correct. This makes ClientID tampering much more difficult. In addition, prior to storing the ClientID in these two locations, the ClientID in each location is reversibly scrambled by the client application with two different keys. This makes it impossible to find the second ClientID location even if someone learns the first location and performs a search based on a value stored in the first location.

### ***ClientID Retrieval Process***

In the beginning of the ClientID retrieval process shown in FIG. 2, the client application attempts to retrieve and unscramble the values stored in both locations. Then it attempts to verify and compare these two values (if any were found).

All possible retrieval outcomes are listed below. Only the first two can be considered "normal", that is, should occur as a part of regular software usage. All other cases indicate that either someone is tampering with the ClientID mechanism or an Operating System malfunction/data corruption has occurred.

- a. ClientID is not found in either of the two locations. This would normally happen when the software is started for the first time on the client machine.  
20 **Action:** request a new ClientID from the backend.
- b. ClientID is found in both locations. The two values have a correct checksum and match each other. This should happen on the second and all subsequent client application launches. **Action:** report retrieved ClientID value to the backend.
- c. ClientID is found in only one location. The value at that location has a correct checksum. **Action:** report retrieved ClientID to the backend along with error code #1 (see below for details)  
25
- d. ClientID is found in both locations. Only one value has a correct checksum. **Action:** report retrieved ClientID from the correct location to the backend along with error code #2 and a value from the other location.
- e. ClientID is found in both locations. The two values have a correct checksum but do not match each other. **Action:** report retrieved ClientID value from  
30

the first location to the backend along with error code #3 and a value from the second location.

f. ClientID is found in both locations. Values from both locations fail the checksum verification. **Action:** request a new ClientID from the backend, report error  
5 code #4 and values from both locations.

g. ClientID is found in only one location. The value at that location fails the checksum verification. **Action:** request a new ClientID from the backend, report error code #5 and a value from that location.

10 In cases c. through g. an error code along with some optional data is reported to the backend. That information is logged on the backend and, in conjunction with other data, like user IP, can be invaluable in detecting fraudulent activity. In cases c. through e. the error code and optional data are stored in the supplied ClientID record. In cases f. and g. that information is stored in the newly generated ClientID record.

15

A number of embodiments of the invention have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the invention. For example, more than two  
20 scrambled versions can be stored in respective locations. Accordingly, other embodiments are within the scope of the following claims.